

DEUXIEME THESE

présentée

A LA FACULTE DES SCIENCES
DE L'UNIVERSITE DE PARIS

pour obtenir

LE GRADE DE DOCTEUR INGENIEUR

par

Radojka DJUKANOVIC

sujet

EVOLUTION DE LA MULTIPROGRAMMATION

Soutenue le 30 Juin 1970 devant la Commission d'Examen

MM. A. BLAQUIERE

Président

Y. LE CORRE

Examineurs

P. DEBRAINE

M. BORY

Invité

DEUXIEME THESE

présentée

A LA FACULTE DES SCIENCES
DE L'UNIVERSITE DE PARIS

pour obtenir

LE GRADE DE DOCTEUR INGENIEUR

par

Radojka DJUKANOVIC

sujet

EVOLUTION DE LA MULTIPROGRAMMATION

Soutenue le 30 Juin 1970 devant la Commission d'Examen

MM. A. BLAQUIERE

Président

Y. LE CORRE

Examineurs

P. DEBRAINE

M. BORY

Invité



1 - INTRODUCTION

On parle couramment de la multiprogrammation pour qualifier le partage dynamique des ressources d'un ordinateur par deux ou plusieurs programmes. Vu de l'extérieur un système de multiprogrammation semble effectuer plusieurs programmes simultanément. Que la simultanéité des opérations de plus d'un programme soit vraie ou apparente, en tout cas le deuxième programme commence avant que le premier soit terminé. La simple coexistence de plusieurs programmes ne suffit pas pour le qualifier de système de multiprogrammation. Il faut qu'il y ait un changement de contrôle constant entre les programmes pour qu'on puisse parler de multiprogrammation.

Avant de discuter de l'évolution de la multiprogrammation nous croyons utile de rappeler les motifs de son développement. Incontestablement, la justification fondamentale d'un système de multiprogrammation repose sur l'avantage économique qu'il offre [1]. Les trois arguments qui expliquent cet avantage sont :

- l'optimisation du rendement de la machine en maximisant l'utilisation de ses éléments
- les réponses en temps réel
- le temps partagé

1) Après les avantages offerts par l'emploi partagé l'amélioration du rendement est actuellement la raison dominante pour utiliser la multiprogrammation et probablement le restera dans un avenir proche. Dans ce contexte, la multiprogrammation implique la possibilité du multitraitement dans le hardware. Si deux ou plusieurs fonctions explicites des programmes ne peuvent pas se dérouler en parallèle, la multiprogrammation ne réalise pas son but concernant l'amélioration du rendements. Pour cette raison on n'a commencé la multiprogrammation qu'après la réalisation des multitraitements élémentaires [2].



Digitized by the Internet Archive
in 2026

https://archive.org/details/IA41549922_0061

La réalisation de l'autonomie des organes d'entrée-sortie (E/S) était très importante pour le développement des calculateurs [3], parce qu'elle a rendu possible le fonctionnement simultané de l'unité centrale et de ces organes. Pourtant la vitesse des organes d'E/S est beaucoup plus petite que celle de l'unité centrale. Une étude statistique portant sur un gros ordinateur a montré que l'unité centrale n'était effectivement employée en moyenne que 65% à 70% du temps de fonctionnement de l'ensemble du système. Le reste du temps, elle attendait que des opérations d'entrée ou de sortie nécessaires à la suite du programme aient été effectuées. Un nouveau problème était posé au programmeur : fournir du travail à l'unité centrale pendant la durée totale des opérations d'E/S.

La solution consiste à loger plusieurs programmes dans la mémoire centrale, le contrôle passant de l'un à l'autre à l'occasion d'une opération d'entrée-sortie, suivant les circonstances, pour garder l'unité centrale en activité. Cette méthode d'exploitation assure l'optimisation des interruptions, évite les pertes de temps et améliore grandement le rendement de l'unité centrale et de sa mémoire centrale, parties les plus coûteuses du système.

2) Il semble que les premiers exemples de multiprogrammation aient apparu avec les systèmes travaillant en temps réel. Dans de nombreuses applications : contrôle des processus industriels, surveillance permanente de l'espace aérien, guidage des avions [4] et des fusées, réservation de places [5] le ordinateur est directement connecté au milieu extérieur qui lui impose son rythme propre. Les programmes à exécuter sont initialisés de l'extérieur. En l'absence de tout ordre extérieur le ordinateur exécute un autre travail défini par un programme appelé programme de fond. Sur l'appel de l'extérieur ce programme est interrompu et le contrôle est donné au programme correspondant à cet appel. Une fois ce programme exécuté, le programme de fond est repris à l'endroit où il avait été interrompu. S'il existe un risque de conflit entre plusieurs appels de l'extérieur il peut être nécessaire d'attribuer à chaque programme un ordre de priorité.

Le programme de fond ainsi que les systèmes d'interruption et de priorité (les éléments de la multiprogrammation) permettent d'occuper le calculateur qui serait autrement sans travail en l'absence de tout ordre extérieur. Le taux d'utilisation des calculateurs est augmenté. Le programme de fond qui remplit les temps morts pourrait être exécuté sur un autre calculateur ce qui entraînerait des frais supplémentaires. Encore une fois, l'intérêt de la multiprogrammation est en premier lieu économique.

3) Les systèmes à temps partagé représentent la forme de multiprogrammation la plus évoluée : l'ordinateur peut être utilisé simultanément par plusieurs programmes travaillant à leur cadence propre, chacun d'eux ayant accès à toutes les ressources de l'ordinateur et ayant l'impression de l'utiliser comme s'il était seul. L'emploi de la multiprogrammation a pour but en donnant successivement le contrôle à divers programmeurs, d'unir suivant un critère économique les possibilités lentes des hommes et les capacités de rapidité du calculateur. Si le prix de revient du calculateur était négligeable, un tel état de fait pourrait se présenter dans l'avenir, il serait tout à fait raisonnable d'attribuer un ordinateur à chaque utilisateur et l'emploi partagé ne serait plus nécessaire. Bien que ce raisonnement néglige certains facteurs importants, tels que l'interaction mutuelle des utilisateurs et le partage des fichiers pendant l'emploi partagé, il illustre la primauté de la motivation économique pour le développement des systèmes de multiprogrammation.

2 - EVOLUTION DE LA MULTIPROGRAMMATION

2.1 - Origine : période 1952 - 1960

On a parlé pour la première fois explicitement de multiprogrammation pour se référer au chevauchement des opérations d'E/S et de l'unité centrale. Dans un rapport publié en 1955 Rochester [2] a exposé un problème de programmation qui nécessitait relativement peu de calculs pour la machine mais un grand nombre d'opérations de lecture et d'écriture. "Le truc utilisé", constatait-il, s'appelle "multiprogrammation" et il consiste à effectuer deux ou plusieurs travaux en même temps... Le contrôleur d'enregistrement sur bande donne l'ordre à l'unité de manipulation de fichier d'explorer l'ensemble des enregistrements. Quand l'unité rencontre un enregistrement actif, une interruption interne a lieu dans l'unité centrale : cette interruption suspend le programme en cours pour traiter cet enregistrement". Dans cette première réalisation on ne pouvait trouver réellement les éléments d'un système de multiprogrammation. Il s'agissait seulement d'une simple suggestion d'utiliser une technique de simultanéité dans un même programme.

Pour permettre une exploitation plus facile et plus générale de la simultanéité des E/S et de l'unité centrale, à cette époque le "truc" était de rendre possible une interruption automatique de programme. Ce développement a éliminé la nécessité d'effectuer des tests longs et difficiles à réaliser pour constater qu'une opération d'E/S était terminée ou qu'il existait une erreur dans le fonctionnement de la machine.

UNIVAC I [6] fut un des premiers calculateur à disposer d'une interruption automatique de programme. Mais dans ce cas, le dépassement de la capacité d'un registre du processeur représentait la seule source qui permettait d'obtenir une interruption. L'extension de la possibilité d'interruption à des instructions d'E/S, extension réalisée sur le calculateur DYSEAC [7], fut un grand pas en avant. A ce sujet LEINER [8] a pu constater : "cela permet d'utiliser de façon optimale la capacité de l'ordinateur en effectuant si-

simultanément plusieurs opérations. En outre le programmeur se trouve libéré de la tâche difficile d'imbriquer les parties asynchrones extérieures et et intérieures, d'un programme."

Une nouvelle extension de la possibilité de l'interruption automatique fut introduite dans le calculateur TX - 2 [9] construit au MIT (Massachusetts Institut of Technology) : quand une interruption se produit le contrôle est donné à l'un des 25 programmes possibles, plus précisément, à celui qui a la plus grande priorité. Le programmeur doit spécifier pour chaque instruction si celle-ci peut être interrompue ou non.

Dans son article publié en 1958 dans le journal britannique "Computer Journal", Gill [10] a proposé un résumé très lucide de l'évolution des problèmes et des possibilités de la multiprogrammation. Son analyse a sans doute beaucoup contribué à l'intérêt croissant que les spécialistes de la programmation ont porté à ce sujet. Son expression "programmation parallèle" remplaçant "multiprogrammation" est à l'origine de l'utilisation de plusieurs synonymes, par exemple : "programmation concurrente", "programmation asynchrone", "programmation simultanée".

Probablement la première et sans doute la plus ambitieuse des tentatives pour introduire des possibilités de multiprogrammation dans un système où les programmes d'application étaient totalement indépendants fut celle qui conduisit à la réalisation du système SHARE 709 (SOS) [11]. SOS était destiné au calculateur IBM 709, machine qui possédait la propriété d'effectuer plusieurs opérations d'E/S en parallèle. L'objectif principal du système d'E/S était de permettre aux programmes d'utiliser cette propriété en demandant un effort minimal au programmeur... Ce système était destiné à traiter des problèmes "scientifiques" pour lesquels les opérations d'E/S n'arrivaient pas dans un ordre fixe". Mock et Swift [11] écrivaient à ce sujet que "pour assurer l'utilisation totale de la capacité de la machine il faut essayer d'effectuer les opérations d'E/S en parallèle avec l'unité centrale plutôt que séquentiellement". Un des objectifs du système SOS était de permettre que soient effectuées en parallèle l'entrée pour l'utilisateur N + 1, le calcul pour l'utilisateur N et la sortie pour l'utilisateur N - 1.

Le pas suivant, très important pour le développement de la multiprogrammation est réalisé par Strachey. Dans son rapport exposé à la conférence ICIP en 1959 [12], il a énoncé très explicitement qu'il était nécessaire d'introduire la notion d'ordre hiérarchique quand plusieurs demandes de temps machine étaient formulées simultanément. L'idée-clé de son travail était de créer un "directeur", c'est un programme spécial qu'on appelle actuellement programme d'ordonnancement (scheduler). Dans l'exemple qu'il proposait Strachey prévoyait que, parmi tous les utilisateurs, un seul était en ligne mais qu'il avait la priorité sur tous les autres. A partir de là, il était possible d'imaginer un système à accès multiple. Si l'on supprime l'utilisateur en ligne, il reste évidemment le prototype d'un système moderne du type "batch processing". Il est possible d'interpréter le travail de Strachey comme une incitation à conférer un caractère plus souple à la planification de l'ordre d'exécution des programmes qui généralement est différent de celui existant dans une queue d'entrée. Ainsi, Strachey est-il à l'origine des travaux qui ont abouti à la conception des systèmes à temps partagé.

Un des articles, certainement les premiers et les plus influents qui traitent l'avenir de la multiprogrammation, est celui présenté par Licklider en 1960 [13]. Son idée de base était de répartir le travail entre l'homme et la machine, ou mieux, de créer une relation symbiotique entre ces deux partenaires, qui permette d'accélérer la résolution des problèmes. Lorsque l'homme et la machine travaillent ensemble les défauts de chacun sont compensés par les qualités de l'autre et les possibilités de cette association sont immenses. Pour la période 1970-75 Licklider a envisagé la création d'un réseau de "centres de réflexion". Ces centres seraient connectés d'une part entre eux, d'autre part à des utilisateurs individuels. "Dans un tel système la vitesse du calculateur serait équilibrée" (le calculateur est trop rapide pour servir un seul utilisateur)" et les frais des mémoires géantes et des programmes sophistiqués seraient partagés par le nombre d'utilisateurs". En résumé, il a conçu un réseau et un système de programmation qui, selon lui, assisteraient l'homme dans son contact avec le calculateur.

Faisant suite aux idées formulées par Strachey, Corbato, Daggett, Daley ont publié probablement le premier article sur les systèmes à temps partagé [14]. Ils assuraient que le système à temps partagé qu'ils proposaient "améliorerait la programmation d'un ou deux ordres de grandeur" et "ouvrirait la voie à plusieurs formes nouvelles d'utilisation du calculateur". Cet article traitait de quelques problèmes, y compris l'organisation des fichiers, la mise au point des programmes, l'ordonnancement. Bien qu'optimistes dans la majeure partie de leur exposé, les auteurs concluaient par l'avertissement : "il est essentiel pour la construction des systèmes futurs que tous les aspects des problèmes relatifs aux systèmes à temps partagé soient explorés et conçus au moyen de prototypes réalisés sur les calculateurs présents de sorte qu'on puissent obtenir une amélioration très importante de l'organisation et l'utilisation des calculateurs."

2.2 - Premières réalisations : période 1961-1964

La période 1961-64 est marquée par l'apparition d'un nombre considérable d'analyses théoriques [15 - 21] sur les systèmes de multiprogrammation et de rapports [22 - 28] résumant les expériences effectuées sur ces systèmes. Toutes ces études aboutissaient à la conclusion qu'une multiprogrammation efficace était réalisable, valable et imminente mais qu'elle imposerait que des constructeurs et des programmeurs atteignent un niveau supérieur dans le domaine de la logique [15]. La conception d'un système de multiprogrammation implique la création coordonnée de trois éléments nouveaux appartenant au hardware et au software :

- logique de contrôle supplémentaire pour faciliter la commutation entre les programmes,
- programme superviseur pour assurer l'enchaînement des programmes présents en mémoire de l'ordinateur,
- système de programmes pour fournir les informations nécessaires à la logique et au programme superviseur (système de priorité des programmes par exemple),

éléments qui sont si étroitement liés qu'il est difficile de dire lequel a le rôle le plus important.

Au lieu de présenter dans ce paragraphe une liste chronologique des réalisations en matière de multiprogrammation, nous avons préféré, pour être plus clair, aborder cette bibliographie en examinant un peu plus en détail l'organisation des systèmes correspondants à cette période.

De tels systèmes possèdent au minimum six éléments :

- systèmes d'interruptions et de priorités ;
- protection mémoires ;
- relocation des programmes et des données ;
- programmes superviseur ;
- adressage symbolique d'E/S ;
- différents modes de fonctionnement des programmes.

2.2.1 - Systèmes d'interruptions et de priorités

Ils constituent, comme nous l'avons déjà vu, les éléments de base parce qu'ils permettent à l'unité centrale de fonctionner avec un rendement acceptable lorsqu'elle est reliée à un grand nombre d'organes périphériques : L'interruption permet de suspendre un programme en cours d'exécution soit pour permettre l'exécution d'un nouveau programme, soit pour indiquer qu'une opération asynchrone est terminée. Ce nouveau programme initialisé par l'interruption effectue les opérations de sauvegarde des registres du calculateur, exécute son propre travail puis restaure l'état général du calculateur. Le contrôle est ensuite rendu au programme interrompu.

La hiérarchie des divers interruptions peut être gérée par programme (par exemple dans les calculateurs ATLAS [21] et Bull GAMMAGO [15], [23]) ou cablé en hardware (le système IBM STRETCH [24, 25]).

Dans le système ATLAS un flip flop change d'état au moment où un signal L. A. M (look at me) se présente. Le programme en cours est alors interrompu et le contrôle est donné au programme d'interruption qui teste

tous les signaux L. A. M pour déceler la cause de l'interruption.

Dans le système STRETCH le système d'interruption fonctionne de la façon suivante : à chaque type d'interruption correspond une ligne et un programme associé. Les interruptions sont hiérarchisées de telle sorte qu'un programme d'interruption ne puisse être interrompu que par une priorité supérieure, tandis qu'un appel de priorité inférieure est mémorisé pour être traité ultérieurement.

Dans le cas d'une hiérarchie cablée des interruptions un certain nombre de tests de contacts permet, après sauvegarde des registres du calculateur, de déceler la cause de l'interruption et de faire exécuter alors le sous programme correspondant.

2.2.2 - Protection de la mémoire

Une caractéristique essentielle de la multiprogrammation est que plusieurs programmes se trouvent logés en même temps dans la mémoire centrale. Ces programmes sont exécutés simultanément (rappelons qu'il s'agit d'une simultanéité apparente). Lorsque l'on désire mettre au point un de ces programmes il se peut que les erreurs éventuelles incluses dans le programme détruisent des zones de mémoire affectées à d'autres programmes. Il est essentiel donc de se protéger contre ce genre d'accident. La protection la plus importante concerne évidemment l'écriture (c'est-à-dire la mise en mémoire d'une donnée) mais il est très utile également de savoir qu'un programme a tenté de lire une donnée ou d'exécuter une instruction dans une zone protégée : de telles initiatives sont des indices d'une mauvaise conception du programme. Il faut être averti de toute violation de la protection et cet avertissement se manifeste par un signal d'interruption.

Il existe plusieurs types de protection de mémoire, par exemple :

a) Une protection par deux registres , supérieur et inférieur, est utilisé dans le système IBM STRETCH. Ces deux registres ont été chargés par le superviseur et ils contiennent les adresses des limites de la zone allouée au programme. Pendant l'exécution de ce programme le hardware compare

les adresses présentes dans les opérendes des instructions avec celles chargées dans les registres. Si une violation est détectée, une interruption se produit.

Le calculateur RCA 601 [26] utilise la même technique mais il est capable d'additionner le contenu du registre inférieur à l'adresse figurant dans le programme. Cette opération permet la relocation dynamique des programmes (au moment de l'exécution).

b) Dans le système ATLAS les programmes d'application et le superviseur ne travaillent pas dans le même mode (cf paragraphe 2.2.6) : le programme d'application génère des adresses sous contrôle du superviseur. Ce dernier utilise un tableau d'adresses associé au programme d'application et autorise ce programme à accéder seulement aux adresses situées dans la zone de la mémoire qui lui appartient.

Pour protéger le programme superviseur le système ATLAS utilise une mémoire spéciale, distincte de la mémoire centrale [22]. Cette mémoire est du type mémoire à lecture seule, inductive ou capacitive. Son contenu ne peut être changé par les programmes d'application.

2.2.3 - Relocation des programmes et des données

Dans un système de multiprogrammation la mémoire centrale n'est pas, en général, assez grande pour contenir tous les programmes. Il n'est pas d'ailleurs nécessaire d'avoir en mémoire centrale tous les programmes car certains ne sont pas utilisés fréquemment. Un programme doit donc pouvoir être amené à tout instant dans la mémoire centrale pour exécution. Puisqu'il est impossible de prévoir l'instant de l'arrivée des programmes et leur durée d'exécution la relocation doit être dynamique : les programmes sont translables c'est-à-dire qu'ils peuvent être exécutés quel que soit l'emplacement où ils vont être implantés. Une partie du programme superviseur, appelé chargeur effectue l'implantation des programmes en mémoire centrale. Il connaît les zones libres de la mémoire et décide à partir de quelle adresse le programme doit être logé.

Nous expliqueront deux types des relocations :

a) Les registres de base sont utilisés dans certains systèmes pour simplifier la relocation. Par exemple, on peut supposer que tous les programmes sont assemblés à partir de l'adresse 0. Au moment du chargement d'un programme le contenu du registre de base est ajouté aux adresses attachées aux instructions.

b) Pagination Le système ATLAS fut le premier de tous les calculateurs à utiliser la relocation dynamique. La pagination est réalisée au niveau du hardware. L'ensemble des mémoires du calculateur ATLAS est divisé en pages. Chaque page contient 512 mots et peut être stockée sur un tambour, un disque ou en mémoire centrale (la mémoire centrale dispose de 32 pages). A chaque page est associé un registre spécial de 11 bits appelé registre d'adresse. Pendant l'exécution d'un programme, le hardware compare chaque adresse figurant dans ce programme avec les contenus de tous les registres d'adresse. Si cette adresse se trouve dans une page appartenant à la mémoire centrale elle est sélectionnée automatiquement, sinon une interruption est générée et le contrôle est donné au superviseur qui recherche la page manquante et l'introduit en mémoire centrale.

Les avantages principaux de la technique de pagination sont les suivants

- il y a protection absolue de la mémoire ;
- l'utilisateur dispose d'un potentiel de mémoire qui peut être beaucoup plus grand que la totalité de la mémoire centrale ;

2.2.4 Programme superviseur

Pour libérer les utilisateurs de la tâche difficile de planifier l'exécution, dans le temps et l'espace, des différents programmes dans un système de multiprogrammation et pour s'assurer que le fonctionnement de ce système est efficace, il est indispensable d'organiser un système de programmes incluant toutes les fonctions d'aide à la multiprogrammation :

- optimisation de l'utilisation des différentes possibilités offertes par le calculateur ;

- gestion des opérations d'E/S ;
- gestion et traitement des interruptions ;
- allocation du temps machine ;
- allocation de la place en mémoire ;

Ce programme s'appelle le superviseur. Il se trouve situé de façon permanente dans une partie protégée de la mémoire. L'organisation hiérarchique du superviseur est représentée sur la figure 1. On voit qu'après le chargement le contrôle est donné à l'exécuteur qui commande toutes les interruptions et tous les sousprogrammes d'erreurs. Tous les programmes d'application sont sous son contrôle. Le sousprogramme d'ordonnancement a fonction de déterminer le prochain travail à exécuter. Dans ce but il se tient au courant des possibilités du système (mémoire, périphériques, unité centrale) et des caractéristiques des programmes sollicitant ces possibilités (ces caractéristiques sont : encombrement en mémoire centrale, volume total, fréquence d'entrées-sorties). Il fait donc une prévision dynamique du travail, prévision qui évolue constamment avec la demande. Il doit tenir compte de toutes les particularités de son environnement de façon à obtenir constamment un bon rendement. Pour cette raison il est difficile d'établir un algorithme général d'ordonnancement. Seule une utilisation du système dans son véritable environnement permet de déterminer l'algorithme qui convient le mieux.

2.2.5 - Adressage symbolique d'E/S

Les unités d'E/S ne peuvent pas être allouées au moment de l'assemblage ou de la compilation. Leur désignation symbolique définie dans un programme d'application est remplacée au moment du chargement par un numéro qui permet le transfert du contrôle vers le sousprogramme chargé d'exécuter les opérations d'E/S.

2.2.6 - Modes de déroulement des programmes

La complexité des systèmes de multiprogrammation et le fait que plusieurs programmes utilisateurs puissent coexister en mémoire centrale, ont conduit à prendre un certain nombre de mesures visant à réduire les possibilités des utilisateurs en leur interdisant certaines fonctions, ceci dans le souci d'éviter qu'ils risquent de perturber la bonne marche de l'ensemble. Ces fonctions sont alors assurées par un programme parfaitement au point : le superviseur.

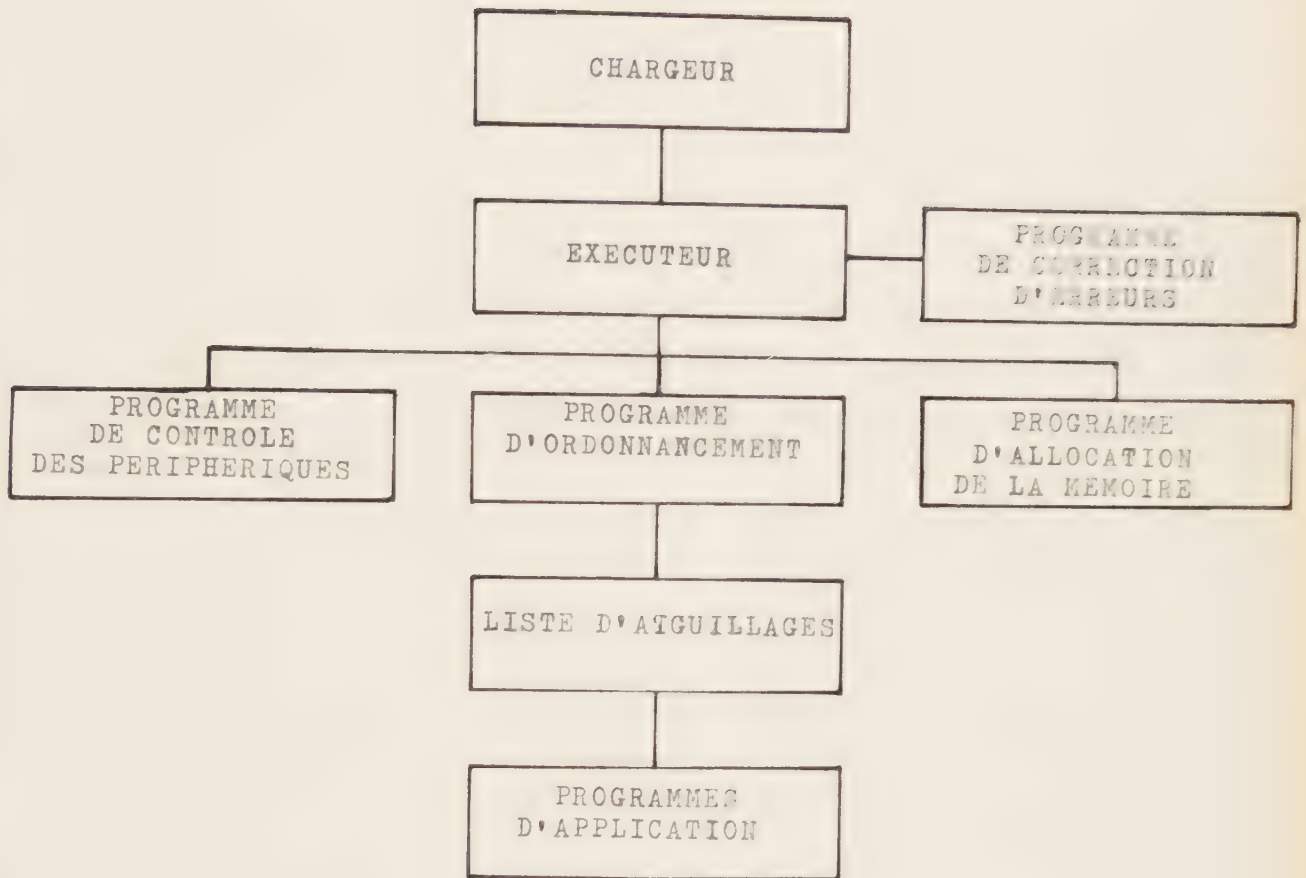


FIGURE 1.

Ceci a conduit à distinguer deux modes appelés généralement : "esclave" et "maître". Le mode esclave est réservé aux programmes des utilisateurs et à certains programmes système comme par exemple le compilateur. Dans ce mode un certain nombre d'instructions sont interdites (entrée-sortie, interruptions, protection mémoire). On les appelle instructions privilégiées.

Le mode maître autorise toutes les instructions.

Les programmes qui constituent le superviseur peuvent être exécutés dans les deux modes. Par contre, lorsqu'un programme utilisateur demande l'exécution d'une opération privilégiée, il le demande au superviseur par un appel particulier. Le superviseur exécute alors l'opération considérée non sans avoir pris toutes les précautions nécessaires quant aux risques d'éventuelles perturbations dans le fonctionnement de l'ensemble du système de multiprogrammation.

L'utilité de deux modes de fonctionnement se justifie pour des raisons de sécurité dans les systèmes de multiprogrammation où la coexistence de programmes non au point est constante.

2.3. Améliorations des systèmes de multiprogrammation période 1965-1969

Durant 1963 et 1964 quelques grands systèmes à temps partagé furent réalisés aux U. S. A. Ils pouvaient répondre à un nombre assez élevé d'utilisateurs. Par exemple le système réalisé sur le calculateur IBM 7090 [27] pouvait servir 24 utilisateurs. Un système plus spécialisé JOSS [28] réalisé pour le calculateur JOHNNIAC pouvait servir environ 100 utilisateurs.

D'intéressantes publications relatives aux améliorations des systèmes de multiprogrammation ont apparu depuis 1965. Les changements proposés les plus importants concernaient les systèmes d'adressage et l'allocation de mémoire.

L'organisation de la mémoire en pages et la translation dynamique, utilisées pour la première fois dans le système ATLAS (cf paragraphe 2.2.3. b)

ont été adoptées par les constructeurs de plusieurs systèmes. On a introduit le terme "mémoire virtuelle" pour définir l'ensemble constitué par les mémoires secondaires (tambours, disques) et la mémoire centrale. La mémoire virtuelle est divisée en pages et les programmes ne sont pas chargés en entier mais par page. L'utilisateur dispose de la totalité de la mémoire virtuelle et puisqu'il n'est plus nécessaire de charger un programme en entier il n'est plus limité par la taille de la mémoire centrale. Le dispositif de la translation dynamique permet le chargement du programme page par page au fur et à mesure des besoins. Le système est organisé de telle sorte qu'au moment de l'exécution chaque adresse logique du programme ou adresse virtuelle est convertie en une adresse réelle. Cette conversion est effectuée automatiquement par la machine si la page est en mémoire centrale à l'instant considéré. Elle sera faite par le système de programmation à la suite d'une interruption programme si la page se trouve dans une des mémoires secondaires. (la page manquante sera d'abord introduite en mémoire centrale). L'adresse physique, réelle qui correspond à l'adresse constante figurant dans l'instruction est une variable qui dépend de l'endroit où la page a été chargée.

A la technique de pagination on a ajouté la technique de segmentation. Cette dernière consiste à diviser la mémoire virtuelle en unités logiques distinctes appelées "segments". Les segments représentent des entités indépendantes ayant leur nom par lequel les utilisateurs peuvent les appeler : Il existe plusieurs types de segments ; citons en deux : programmes et données. Un programme utilisateur est en général composé de plusieurs segments qui d'autre part peuvent être partagés par plusieurs utilisateurs sans être modifiés pendant les différentes exécutions ; les segments constituent donc des espaces indépendants, appelés des "espaces communs". Le but de la segmentation est de permettre d'utiliser de la manière la plus générale ces espaces communs" à plusieurs utilisateurs. En termes plus concrets, cela signifie l'utilisation simultanée d'un programme ou d'un bloc de données par plusieurs processus indépendants.

La notion de segment fut introduite par Holt [32] et développée par Dennis [33], Forgie [34] et Greenfield [35].

Les premiers systèmes liés à cette nouvelle conception (pagination, mémoire virtuelle, translation dynamique, segmentation) ont été conçus pour des calculateurs d'une très grande capacité. GE [29, 30], IBM 360/67 [36, 37] et BORROUGHS 5000 [38].

Une nouvelle technique d'adressage est associée aux calculateurs IBM 360/67 et GE 645. C'est celle des mémoires associatives ou mémoires adressables par leur contenu. Cette technique accélère notablement le temps d'accès aux mémoires, ce qui réduit le temps d'exécution des programmes et améliore les performances des systèmes.

Les systèmes IBM 360/67 et GE 645 se sont distingués par une autre nouveauté ; ce sont des systèmes multiprocesseurs, c'est-à-dire organisés autour de plusieurs unités de traitement identiques pouvant fonctionner en simultanéité et accéder aux mêmes blocs de mémoire.

Ces deux systèmes disposaient ainsi d'une puissance et d'une fiabilité fortement accrues.

3 - CONCLUSION

Le développement des techniques du calcul en temps réel et en temps partagé fut très rapide et les possibilités dans ces domaines sont devenues si grandes qu'il est impossible même de les envisager toutes à l'heure actuelle.

C'est une banalité de mentionner que tous les progrès techniques ont été acquis aux Etats-Unis. Les moyens matériels et une large expérience dans l'emploi partagé y ont créé un milieu privilégié où les audaces techniques deviennent un fait quotidien. L'Europe "suit" avec quelques années de décalage. Suivant le même processus qu'outre-Atlantique ce sont principalement les Universités qui font les premiers pas. Des équipes de chercheurs et d'étudiants ont déjà réalisé des systèmes de multiprogrammation.

Pourtant une diffusion commerciale plus large dans un contexte de rentabilité et d'utilité plus immédiates est encore hésitante. Après une assez longue période d'attente les principaux constructeurs de calculateurs semblent se préparer à franchir le pas et à amorcer la prochaine phase qui sera caractérisée par la mise en service d'un réseau d'abonnés utilisant le temps partagé. Les terminaux seront les ordinateurs satellites, systèmes en temps réel (système de réservation de places, réseaux bancaires, banques d'informations). De tels systèmes existent déjà mais ce sont des systèmes expérimentaux. Beaucoup de problèmes restent à régler pour que ces systèmes deviennent réellement économiques. La prochaine étape sera caractérisée par une évolution continue dans l'ensemble hardware + software + firmware afin de diminuer le prix et d'améliorer les performances des systèmes de multiprogrammation.

Un des aspects les plus intéressants des systèmes de multiprogrammation est la possibilité d'accès à une vaste collection de programmes et de données. Dans ce but la mémoire centrale est couplée à une mémoire de masse d'une grande capacité, très lente, à accès aléatoire, où sont stockées les don-

nées et la bibliothèque des programmes moins fréquemment utilisés.

De nouvelles techniques de stockage de masse sont l'objet des recherches actuelles. Les films minces, par exemple où la densité de stockage est de l'ordre de 1 bit par millimètre et la rapidité de 300 nanosecondes ouvrent la possibilité de doter les machines de mémoires de plusieurs millions octets avec un cycle d'environ 600 nanosecondes ce qui permettrait d'employer des méthodes et des techniques d'utilisation et de programmation moins conventionnelle, d'une efficacité très supérieure.

Le programme superviseur qui était conçu pour améliorer le rendement de la machine et du service rendu à l'utilisateur pose des problèmes considérables à ses constructeurs. Puisqu'il doit être très sophistiqué et général, il devient très complexe et difficile à comprendre. Il est aussi presque impossible à changer même si le changement désiré concerne une des parties très spécialisées du superviseur. D'autre part le superviseur est très cher parce qu'il nécessite beaucoup d'espace pour son logement et de temps pour effectuer ces nombreuses tâches.

Il existe déjà les moyens pour diminuer la place occupée par le programme superviseur ainsi que pour améliorer le temps d'exécution de ce programme. Dans les machines de la 4ème génération (on prévoit qu'elle commenceront à être livrées vers 1972/1975) l'unité centrale sera considérablement plus rapide que la mémoire centrale. Ceci permettra à l'unité centrale d'effectuer plusieurs opérations avant que la nouvelle instruction (commande), en provenance de la mémoire centrale se présente. C'est le "firmware" qui intervient alors : [41, 42] il est constitué par l'enregistrement sous forme de microprogramme de séquences élémentaires composant chaque instruction, cet enregistrement étant effectué dans une mémoire très rapide, mémoire à lecture seule. Chaque enregistrement correspond à une opération à effectuer et l'ensemble des enregistrements correspondant à une instruction, représente un petit sousprogramme. Le sousprogramme est initialisé par la

commande arrivant de la mémoire centrale. Puisque ces sousprogrammes peuvent être changés (par l'intermédiaire d'un système de lecture de cartes magnétiques spéciales) et qu'ils seront la base du software le nom "firmware" ("firm", c'est-à-dire entre "hard" et "soft") décrit exactement leurs places dans la technologie d'un ordinateur.

L'utilisation du firmware s'étendra surtout au niveau des programmes superviseur : les compilateurs et les programmes d'exécution pourront être bâtis au sein du firmware ce qui réduira l'espace occupé par ces programmes et améliorera les temps de compilation et d'exécution. Le fonctionnement des ordinateurs utilisés dans les systèmes de multiprogrammation sera grandement accéléré.

REFERENCES

- [1] CODD E. F.
Multiprogramming
Advances in computers, vol 3. Academic Press,
New York 1962, p. 77
- [2] ROCHESTER N.
The computers and its peripheral equipment
Proc EJCC, 1955, p. 64
- [3] MARCOTTY M. J., LONGSTAFF F. M., WILLIAMS A. P. M.
Time sharing on the FERRANTI-PACKAPD FP 6000
computer system
Proc SJCC, 1963, p. 29
- [4] EVERETT R. R., ZRAKET C. A., BENNINGTON H. D.
SAGE - a data processing system for air defense
Proc EJCC, 1957, p. 148
- [5] PARKER R. W.
The SABRE system
Datamation, September 1965, p. 49
- [6] ECKERT J. P., WEINER J. R., WELSH H. F., MITCHELL H. F.
The UNIVAC system
Proc. Joint AIEE-IRE Conf., February 1952, p. 6
- [7] LEINER A. L., ALEXANDER S. N.
System organization of DYSEAC
IRE Trans on Electronics Computers, EC-3, 1954, p. 1

- [8] LEINER A. L.
System specification for the DYSEAC
JACM, 1, 1954, p. 57
- [9] FRANKOVICH J. M., PETERSON H. P
A functional description of the Lincoln TX-2 computer
Proc. WJCC, 1957, p. 146
- [10] GILL S.
Parallel programming
Computer J., 1, 1958, p. 2
- [11] MOCK O., SWIFT C. J
The SHARE 709 System programmed input output buffering
JACM, 6; 1959, p. 145
- [12] STRACHEY C.
Time sharing in large fast computers
Proc ICIP, 1959, p 347
- [13] LICKLIDER J. C. R
Man computer symbiosis
IRE Trans on human factors in Electronics, HFE-8, 1960, p. 4
- [14] CORBATO F. J., DAGGETT M. M., SALEY R. C.
An experimental time sharing system
Proc SJCC, 1962, p. 335
- [15] RYLE B. L.
Multiple programming data processing
CACM, February, 1961, p. 99
- [16] AMDAHL G. M.
New concepts in computing system design
Proc IRE, May 1962, p. 1073

- [17] TURNER L. R., RAWLINGS J. H.
Realization of randomly timed computer input and
output by means of an interrupt feature.
IRE Trans. on Electronic Computers, EC-7, 1958, p. 141
- [18] KILBURN T. , EDWARDS D. B. G. , LANIGAN M. J. , SUMNER F. H.
One level storage system
IRE Trans on Electronic Computers, EC-11, 1962, p. 223
- [19] WEIL J. W
A heuristic for page turning in multiprogramming computer
CACM, September 1962, p. 480
- [20] LANDS N. , MANOS A. , TURNER L. R.
Initial experiences with an operating multiprogramming system
CACM, May 1962
- [21] KILBURN T. , HOWARTH D. J. , PAYNE R. B. , SUMNER F. H.
The Manchester university Atlas operating system,
Part I : Internal organization
Computer J. , 4, 1961, p. 222
- [22] FORTHERINGHAM J.
Dynamic storage allocation in the Atlas computer including an
automatic use of a backing store
CACM, April 1962, p. 435
- [23] DREYFUS P.
System design feature of the GAMMA 60 computer
Proc WYCC, 1958, p. 130
- [24] BROOKS F. P. J. R.
A program controlled program interruption system
Proc EJCC, 1957

- [25] CODD, E.F., LOWRY E.S., MCDONOGH E., SCALZI C.A.
Multiprogramming STRETCH : feasibility considerations
CACM, November 1959, p. 13
- [26] SMITH R.D.
Multiprogramming the RCA 601
Preprints of papers presented at the 16th National
Meeting of the ACM, 1961, p. 12C-1
- [27] KINSLOW H.A
The time sharing monitor system
Proc. FJCC, 1964, p. 443.
- [28] JOSS : a designer's view of an experimental on line
computing system
Proc FJCC, 1964, p. 445
- [29] CORBATO F.J., VYSSOTSKY V.A.
Introduction and overview of the MULTICS system
Proc. FJCC, 1965, p. 185
- [30] VYSSOTSKY V.A. , CORBATO F.J.
Structure of the MULTICS supervisor
Proc FJCC, 1965, p. 203
- [32] HOLT A.W.
Program organisation and record heeping in dynamic storage allocation
CACM, October 1961, p. 422
- [33] DENNIS J.B
Segmentation and the design of multiprogrammed
computer systems
JACM, 4, 1965, p. 589

[34] FORGIE J. W.

A time and memory sharing executive program for
quick response on line application.

Proc FJCC, 1965, p. 599

[35] GREENFIELD M. N.

Fact Segmentation

Proc SJCC, 1962, p. 307

[36] COMFORT W. T.

A computing system design for user service

Proc FJCC, 1965, p. 619

[37] EVANS D. C., LECLERC J. Y.

Address mapping and the control of acces in an
interactive computer

Proc SJCC, 1967, p. 23

[38] MACKENZIE F. B.

Automated secondary storage management

Datamation, November 1965, p. 24

[39] SHWARTZ J. I. , WEISMAN C.

A general purpose time sharing system

Proc SJCC, 1964, p. 397

[40] SHWARTZ J. I. , WEISMAN c. , COFFMAN E. G.

The SDC time sharing system revisited

Proc of the 22 ud National ACM Conf., 1967, p. 263

[41] OPLER A.

Forth- generation software

Datamation, Jamary 1967, p. 22

[42] JEWETT D. J.

A forth generation computer now?

Datamation, April 1968, p. 37

[43] BERTIN J., RITOUT M., ROUGIER J. C.

L'exploitation partagé des calculateurs

Dunod, Paris, 1967

[44] BOLLIET L.

Utilisation des ordinateurs à distance en temps réel et en temps
partagé

Dunod, Paris, 1967

[45] MARTIN J.

Programming real time computer system

Prentice Hall, 1965.

